

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

## REMARKS

The Office Action of February 27, 2003, has been carefully considered. Claims 1 - 18 are pending in the application. Claims 1-18 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,903,894 to Reneris (hereinafter Reneris). Claims 1 and 7 have been amended. The above rejections have been traversed in view of the following remarks that include a brief description of certain aspects of the invention and a brief description of the teachings in the cited art.

## I. REJECTIONS

### 35 U.S.C. § 102 Claim Rejections

Claims 1-18 were rejected under 35 U.S.C. § 102(b) as being anticipated by Reneris. In review, in order for prior art to anticipate a claim under 35 U.S.C. § 102 every element of the claimed invention must be identically disclosed either expressly or under principles of inherency in a single reference. Further, the exclusion of a claimed element from a prior art reference, no matter how insubstantial, is enough to negate anticipation by that reference. The test of whether anticipation exists in a particular case is a question of fact, and is applied element-by-element to a single prior art reference. Only if the prior art literally reads on every element of the rejected claim will the claimed invention be anticipated under this test.

The present invention is directed at a method for controlling PCI devices via firmware without requiring manufactures of relatively simple PCI devices to create device-specific drivers. As explained in the background section, in the past, because the operating system expected to be able to use every PCI device that it detected, if the PCI device did not have a driver available, the operating system treated this as an error situation. However, requiring hardware manufactures to create a device driver for each device was burdensome because certain types of devices were too simply and did not justify a device driver. Page 2, lines 15-20. In addition, because the operating system could dynamically reconfigure any PCI device within the computer's address space, firmware was unable to deterministically read from or write to PCI devices. Page 2, line 21 to Page 3, line 5. The operating system also assigned ownership of address space for the PCI device to a specific device driver and only that device driver could write to or read from that

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

assigned address space. All of these factors prevented computer manufactures from supporting firmware-controlled devices. Page 3, lines 5-14.

With this problem in mind, the present invention enables a PCI BAR Operation Region through which PCI devices can be accessed. Firmware is used to control the device. The firmware describes the PCI device using AML (ACPI Machine Language). The AML declares a PCI BAR operation region associated with the PCI device. A generic device driver is loaded and is registered to handle accesses to or from the PCI device via the PCI BAR operation region. The generic driver stores the bases addresses assigned to each PCI device on the PCI bus. When the firmware accesses the PCI BAR operation region by identifying the PCI BAR number and offset, the generic driver resolves that information into an absolute memory or I/O address based on the current BAR assigned. Thus, with the present invention, manufactures may continue providing firmware for simple PCI devices without having to create a device-specific driver. Page 3, line 16 to Page 4, line 18.

Reneris is directed at a system and method for using a hierarchical data structure to control and identify devices and represent connections between devices. Reneris is directed at solving the following problems: 1) describing the hardware devices to the operating system; 2) describing connections or dependencies between the devices; and 3) providing extensible and abstract methods of controlling the devices executable by the operating system or any other software module, such as the BIOS routines. Col. 2, lines 8-14. Reneris teaches to specify a first and second object in a hierarchical data structure, with the second object positioned immediately below the first object to form a hierarchy between the first object and the second object. The hierarchy then accurately represents the physical input/output connection between the first device and second device. This allows the operating system to determine which of a group of similar devices exist at different locations within the computer system have changed configuration, have hardware events related to them, or have to be controlled. Col. 2, lines 16-38.

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

### **Independent Claim 1 and its dependent Claims 2-6**

Rejected independent Claim 1, from which rejected Claims 2-6 depend, now recites "the process provider being programmed to resolve and maintain addressing information for a plurality of PCI devices, including the PCI device." As briefly summarized above, the present invention does not require a unique device driver for each PCI device. Rather, the present invention enables a "process provider" that is registered to handle the operation region associated with all simple PCI devices. Page 16, lines 25-26. In one implementation, a special "PCI BAR target provider driver" 421 shown in FIGURE 4 is used. In another implementation, an ACPI driver 202 (FIGURE 2) registers itself to handle the PCI devices. Page 20, line 7. In either case, for these types of PCI devices (PCI device 241 shown in FIGURE 2) there is no special device driver to control it. Page 19, line 25-26. Instead, the PCI device is firmware-controlled rather than operating system controlled. Page 19, line 26 to Page 20, line 1. Thus, these devices are controlled by a "process provider being programmed to resolve and maintain addressing information for a plurality of PCI devices", as recited in Claim 1.

In contrast, Reneris teaches that the operating system loads an appropriate device driver for each of the identified devices. The appropriate device driver supports one of the specific device objects. The device driver is a conventional software module used by the operating system to access devices connected to the computer system. As further taught by Reneris, typically, the Original Equipment Manufacture (OEM) who has supplied the associated device provides the device driver. Thus, the device driver provides the operating system with the functionality of the device. Col. 19, lines 30-39. The hierarchical data structure taught by Reneris allows additional functionality for a device by allowing the device to "inherit" functionality provided by another device above it in the hierarchy. Col. 19, lines 51-57. However, even for this additional functionality, the functionality was first specified in a specific device driver associated with a particular device.

Thus, Reneris neither teaches nor suggests having a process provider that resolves and maintains addressing information for a plurality of PCI devices as recited in Claim 1. In particular, Figures 3A-3B and column 12, lines 22-23, cited by the Examiner for teaching the use

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

of a process provider merely recite that in step 340 the ACPI driver reads a static block of memory in the FADT (Fixed ACPI Description Table) that describes hardware control registers and I/O space for the system. Then, the ACPI driver initializes hardware control registers and I/O space. Interestingly, Reneris does not even mention that the operating system may change the addresses for these control registers and I/O space. Therefore, Reneris would not even encounter some of the problems that the present invention overcomes, namely that a plug and play operating system may change the values of these base address registers at any time to relocate the devices within the computer address space. Thus, having a static block of memory that describes hardware control registers and I/O space would not work, and definitely does not teach or suggest having a process provider programmed *to resolve and maintain addressing information for a plurality of PCI devices*.

In addition, dependent Claims 2-6 include other limitations that are not taught or suggested by Reneris. Therefore, for at least the above arguments, Applicant respectfully submits that the § 102(b) rejections of Claims 1-6 are improper, and respectfully requests reconsideration and withdrawal of these rejections.

#### **Independent Claim 7, and dependent Claims 8-18**

The Claims 7-18 were rejected under 35 U.S.C. § 102(b) as being anticipated by Reneris. Rejected independent Claim 7, from which rejected Claims 8-18 depend, now recites "a PCI object that defines an operation region for accessing a firmware-controlled PCI device" and "the operation region facilitating firmware-controlled interaction with the PCI device." As explained in the specification of the present invention and above, certain type of devices are quite simple and do not justify that a specific device driver be written for the device. However, in the past, in order to use the Peripheral Component Interconnect (PCI) bus, the devices had to have a unique device driver. Therefore, the present invention provides a PCI object in which an operation region is defined for accessing a firmware-controlled PCI device.

Without repeating the arguments from above, in summary, Reneris teaches that the operating system loads an appropriate device driver for each of the identified devices. Reneris

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

does not teach or even suggest that devices may not have a device driver associated with it. In addition, Reneris definitely does not teach or suggest using firmware-controlled devices. As described in the specification of the present application, the PCI device of the present invention is firmware-controlled rather than operating system controlled. Page 19, line 26 to Page 20, line 1.

In addition, Reneris neither teaches nor suggests "a definition block including a PCI object that defines an operation region for accessing a firmware-controlled PCI device" as recited in Claim 7. Rather, the operation region taught in Reneris is described as follows:

The operation region of the device essentially is an abstract manner of representing a hardware control register, I/O space, or another controllable element of the device. One skilled in the art will understand that the "operation region" is defined as a pseudocode language construct that provides the abstract manner of accessing the controllable elements of the device. Col. 13, lines 30-43.

Reneris further explains the use of the operational region as follows:

Access to the actual device is accomplished by reading and writing to a special data object (operational region) representing a controllable element of the device, such as a control register or I/O space of the device. As previously mentioned, these special data objects provide an abstract, yet precise access down to the actual bit level, depending on how the special data object is defined. Col. 14, lines 22-28.

Reneris also explains how the special data object (operational region) obtains its information as follows:

The operating system loads an appropriate device driver for the identified device to support the corresponding device object. Col. 19, lines 30-32.

In summary, Reneris teaches that each device has an associated device driver from which a corresponding device object is created. The device object being the "special data object (operational region)." Thus, the operation region taught in Reneris is directly associated with having a traditional device driver for each device, which is controlled by the operating system. In contrast, the present invention is directed at firmware-controlled PCI devices. The firmware-controlled PCI devices do not have a special device driver associated with it. Thus, Reneris

App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

cannot possibly teach or suggest "a firmware-controlled PCI device" as recited in Claim 7. Furthermore, Reneris does not teach or suggest an "operation region facilitating firmware-controlled interaction with the PCI device" as recited in Claim 7 of the present application.

In addition, dependent Claims 8-18 include other limitations that are not taught or suggested by Reneris. Therefore, for at least the above arguments, Applicant respectfully submits that the § 102(b) rejections of Claims 7-18 are improper, and respectfully requests reconsideration and withdrawal of these rejections.

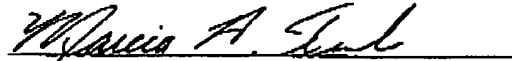
App. No. 09/809,639  
Amendment Dated July 25, 2003  
Reply to Office Action of February 27, 2003

## II. CONCLUSION

Applicant has considered the other references cited by the Examiner in the Office Action. None of these references appear to affect the patentability of Applicant's claims. By the foregoing remarks, Applicant believes that all pending claims are allowable and the application is in condition for allowance. Therefore, a Notice of Allowance is respectfully requested. Should the Examiner have any further issues regarding this application, the Examiner is requested to contact the undersigned attorney for the Applicant at the telephone number provided below.

Respectfully submitted,

MERCHANT & GOULD P.C.



Marcia A. Tunheim  
Registration No. 42,189  
Direct Dial: 206.342.6259

MERCHANT & GOULD P.C.  
P. O. Box 2903  
Minneapolis, Minnesota 55402-0903  
206.342.6200